CLAIMS

1. A method for dynamic allocation of memory address space, comprising:

protecting an optimized version of a program from modification;

executing an original version of the program, the executing of the original version of the program including executing a request to use memory address space occupied by the optimized version of the program;

copying a portion of the optimized version of the program residing in the memory address space requested by the original version of the program, writing the copied portion of the optimized version of the program to unallocated memory address space, and adjusting code of the optimized version of the program;

releasing the protecting of the copied portion of the optimized version of the program from modification after completing the writing;

returning execution control to the original version of the program; and

re-executing the request to use the memory address space occupied by the portion of the optimized version of the program for which the protecting has been released.

2. The method of claim 1, wherein the optimized version of the program is defined by pages of code.

3. The method of claim 1, wherein the portion of the optimized version of the program that is copied and written to unallocated memory address space is defined by one or more pages of code.

4. The method of claim 1, wherein the memory address space to which the copied portion of the optimized version of the program is written is protected from modification.

5. The method of claim 4, wherein the memory address space that is protected from modification is defined by one or more protected pages of code.

6. The method of claim 1, wherein the adjusting of the code of the optimized version of the program is configured to enable interoperability of the code of the optimized version of the program.

7. The method of claim 1, wherein the adjusting of the code of the optimized version of the program includes adjusting address data associated with at least one of pointers, branch instructions, and jump instructions.

8. The method of claim 1, wherein the protecting of the optimized version of the program places the optimized version of the program in a read-only state.

9. The method of claim 8, wherein the releasing of the protecting of the optimized version of the program removes the read-only state.

10. The method of claim 1, further comprising:

detecting the request to use the memory address space occupied by the optimized version of the program during execution of the original version of the program; and

passing execution control to an optimization code that was used to define the optimized version of the program, the optimization code being configured to execute the copying and the releasing.

11. The method of claim 1, wherein the protecting is carried out by a hardware device.

12. A method for dynamic allocation of memory address space, comprising:

executing an original version of a program, the executing of the original version of the program including executing a request to use memory address space occupied by an optimized version of the program, the optimized version of the program being protected from modification;

detecting the request to use the memory address space occupied by the optimized version of the program during execution of the original version of the program;

passing execution control to an optimization code that was used to define the optimized version of the program, the optimization code executing

copying of a portion of the optimized version of the program residing in the memory address space requested by the original version of the program, writing of the copied portion of the optimized version of the program to unallocated memory address space, and adjusting of code of the optimized version of the program; and

releasing of the protecting of the copied portion of the optimized version of the program from modification after completing the writing;

returning execution control to the original version of the program; and

re-executing the request to use the memory address space occupied by the portion of the optimized version of the program for which the protecting has been released.

13. The method of claim 12, wherein the optimized version of the program is defined by pages of code.

14. The method of claim 12, wherein the portion of the optimized version of the program that is copied and written to unallocated memory address space is defined by one or more pages of code.

15. The method of claim 12, wherein the memory address space to which the copied portion of the optimized version of the program is written is protected from modification.

16. The method of claim 15, wherein the memory address space that is protected from modification is defined by one or more protected pages of code.

17. The method of claim 12, wherein the adjusting of the code of the optimized version of the program is configured to enable interoperability of the code of the optimized version of the program.

18. The method of claim 12, wherein the adjusting of the code of the optimized version of the program includes adjusting address data associated with at least one of pointers, branch instructions, and jump instructions.

19. The method of claim 12, wherein the protecting of the optimized version of the program places the optimized version of the program in a read-only state.

20. The method of claim 19, wherein the releasing of the protecting of the optimized version of the program removes the read-only state.

21. The method of claim 12, wherein the protecting is carried out by a hardware device.

22. Computer readable media containing program instructions for dynamic allocation of memory address space, the computer readable media comprising:

program instructions for protecting an optimized version of a program from modification;

program instructions for executing an original version of the program, the executing of the original version of the program including executing a request to use memory address space occupied by the optimized version of the program;

program instructions for copying a portion of the optimized version of the program residing in the memory address space requested by the original version of the program, writing the copied portion of the optimized version of the program to unallocated memory address space, and adjusting code of the optimized version of the program;

program instructions for releasing the protecting of the copied portion of the optimized version of the program from modification after completing the writing;

program instructions for returning execution control to the original version of the program; and

program instructions for re-executing the request to use the memory address space occupied by the portion of the optimized version of the program for which the protecting has been released.

23. The computer readable media of claim 22, wherein the optimized version of the program is defined by pages of code.

24. The computer readable media of claim 22, wherein the portion of the optimized version of the program that is copied and written to unallocated memory address space is defined by one or more pages of code.

25. The computer readable media of claim 22, wherein the memory address space to which the copied portion of the optimized version of the program is written is protected from modification.

26. The computer readable media of claim 25, wherein the memory address space that is protected from modification is defined by one or more protected pages of code.

27. The computer readable media of claim 22, wherein the adjusting of the code of the optimized version of the program is configured to enable interoperability of the code of the optimized version of the program.

28. The computer readable media of claim 22, wherein the adjusting of the code of the optimized version of the program includes adjusting address data associated with at least one of pointers, branch instructions, and jump instructions.

29. The computer readable media of claim 22, wherein the protecting of the optimized version of the program places the optimized version of the program in a read-only state.

30. The computer readable media of claim 29, wherein the releasing of the protecting of the optimized version of the program removes the read-only state.

31. The computer readable media of claim 22, further comprising:

program instructions for detecting the request to use the memory address space occupied by the optimized version of the program during execution of the original version of the program; and

program instructions for passing execution control to an optimization code that was used to define the optimized version of the program, the optimization code being configured to execute the copying and the releasing.

32. The computer readable media of claim 22, wherein the protecting is carried out by a hardware device.

33. A hardware component including logic for dynamic allocation of memory address space, comprising:

logic for executing an original version of a program, the executing of the original version of the program including executing a request to use memory address space occupied by an optimized version of the program, the optimized version of the program being protected from modification;

logic for detecting the request to use the memory address space occupied by the optimized version of the program during execution of the original version of the program;

logic for passing execution control to an optimization code that was used to define the optimized version of the program, the optimization code executing

copying of a portion of the optimized version of the program residing in the memory address space requested by the original version of the program, writing of the copied portion of the optimized version of the program to unallocated memory address space, and adjusting of code of the optimized version of the program; and

releasing of the protecting of the copied portion of the optimized version of the program from modification after completing the writing;

logic for returning execution control to the original version of the program; and

logic for re-executing the request to use the memory address space occupied by the portion of the optimized version of the program for which the protecting has been released.

34. The hardware component of claim 33, wherein the optimized version of the program is defined by pages of code.

35. The hardware component of claim 33, wherein the portion of the optimized version of the program that is copied and written to unallocated memory address space is defined by one or more pages of code.

36. The hardware component of claim 33, wherein the memory address space to which the copied portion of the optimized version of the program is written is protected from modification.

37. The hardware component of claim 36, wherein the memory address space that is protected from modification is defined by one or more protected pages of code.

38. The hardware component of claim 33, wherein the adjusting of the code of the optimized version of the program is configured to enable interoperability of the code of the optimized version of the program.

39. The hardware component of claim 33, wherein the adjusting of the code of the optimized version of the program includes adjusting address data associated with at least one of pointers, branch instructions, and jump instructions.

40. The hardware component of claim 33, wherein the protecting of the optimized version of the program places the optimized version of the program in a read-only state.

41. The hardware component of claim 40, wherein the releasing of the protecting of the optimized version of the program removes the read-only state.

42. The hardware component of claim 33, wherein the hardware component is a processor.